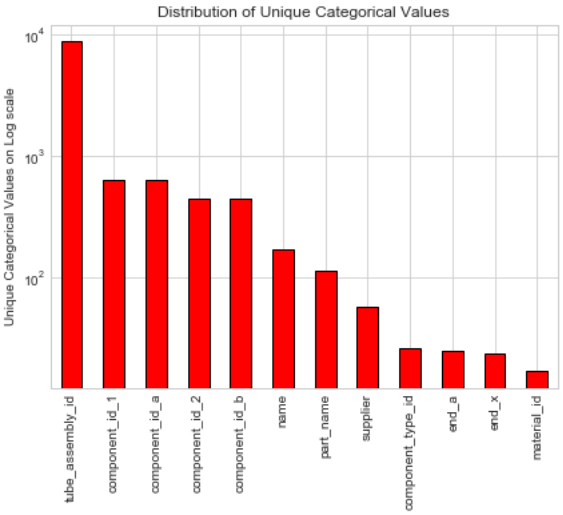
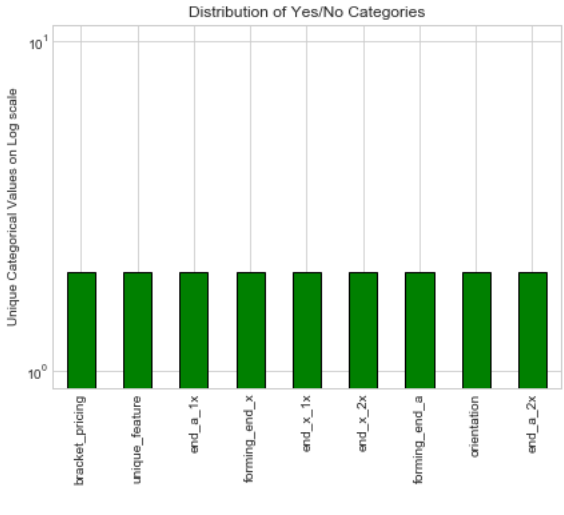
Data Science Capstone Project 1
Model Selection and Evaluation

Part 1- Supplier Pricing Prediction

After pre-processing data and conducting exploratory data analysis, the final table contained two types of categorical values as shown in the graph below:

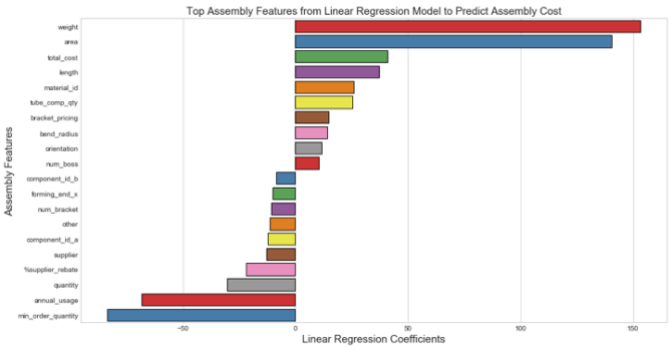
Ordinal represented in red	Nominal represented in Green
	

It was important to convert these values into binary values (0 ,1) so that model can also learn from categorical features. I used label-encoding (Model A) and binary encoder (Model B) to train and evaluate linear and non-parametric models.

To convert categorical features, I tried one-hot encoding with label encoder but it resulted in a sparse matrix with more than 1000 + features, causing curse of dimensionality. Then tried label encoder without one-hot encoder a.k.a Model A which reduced column labels down to 36 but caused
ordinal issues in the model. Finally, Binary Encoder a.k.a Model B seemed to be the best choice because it reduced 1000+ down to 92 as well as managed ordinal issues by improving model performance.

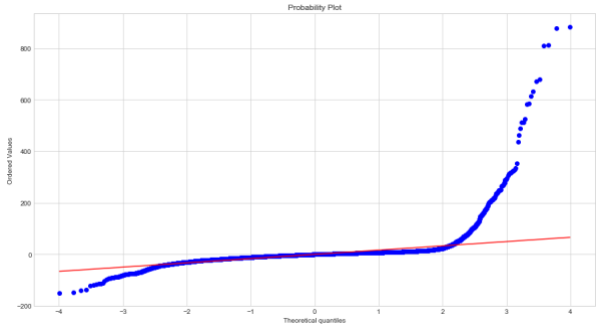
Applied Models and Evaluation:

1. Linear Regression using Model A: This is one of the simplest supervised learning models which used a set of independent input variables (x) to predict a linearly dependent outcome (y), represented by an equation y = mx +c.
   Here m and c are the coefficients, y is the supplier price we are trying to predict, x are the input features specifications, order history, demand and more. I, applied this model to uncover the relationship between tube assembly variables and assess which features can be used to predict supplier pricing.



As seen in the graph, we uncovered linearly positive variables such as weight, area, total cost and linearly negative variables such as supplier rebates, quantity and usage as top features in predicting supplier price with train-test evaluation score of 31% to 32%.

However, this model also showed right-skewed residual distribution plot, indicating strong presence of non-linearity in the dataset, making it not suitable for this use case.

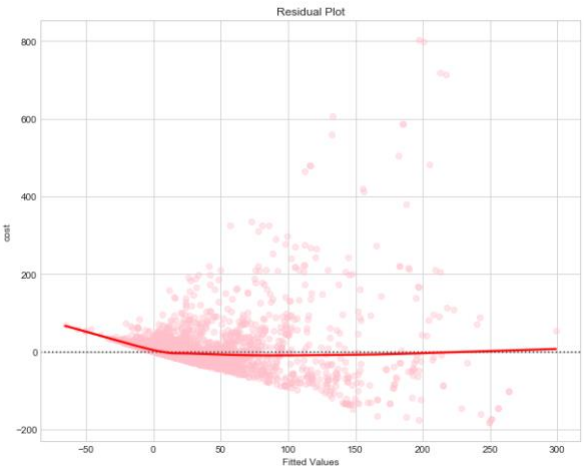


If we refer to Quantile plot, it is evident that residual contains multiple outliers in the dataset which is either above or below the redline but not directly on the straight redline as shown in the graph.

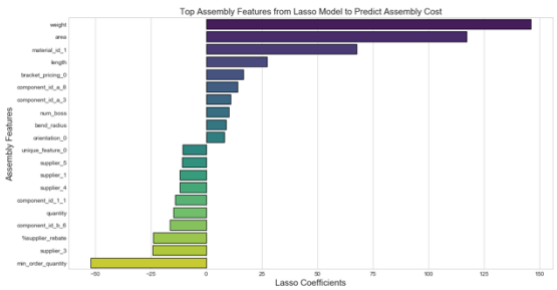
So, the main question arises, if we removed residual outliers, would this linear model work?

The answer is No. After removing outliers and influence points (outliers with maximum impact on linear coefficients), R2 score (explained variance) for training and test predictions resulted in 100% accuracy, causing the model to overfit.

2. **Linear Regression using Model B:** This time, by applying binary encoder, the same model Linear Regression model which we used above, resulted in higher R2 score (40% to 42%) for training and test model predictions, accounting for 31.25% improvement over Model A. However, residual distribution continued to show right-skewness making both models not a good selection for predicting assembly cost.



3. **Lasso Regression using Model B:** This model reduced complexity of linear models by only keeping fewer features which had non-zero coefficients and were non-collinear. I applied a tuning parameter 'alpha' which penalized the sum of absolute values of the coefficients, thereby rejecting coefficients that were zero value or less important in making predictions



I, used this model to identify important features and see how it would compare with previous models.

I observed that the top three features weight, area and minimum order quantity maintained its position ranking but other features such as material id, length, contract pricing had gained higher preference in feature selection.

R2 score improved from 32% to 41% and predictors such as weight and minimum order quantity showed 4.57% and 38% decrease in assembly cost with one-unit increase, respectively.

5. **Experiment (Linear Regression on Aggregate Table using Model A):** Out of curiosity, instead of using individual points, I wanted to build a summary table containing mean and count of numerical and categorical features. This, however, did not perform well and resulted in 11.2% difference between training and test model evaluation but on the positive note, gave lower log RMSE value.

6. **Decision Tree Regressor:** This model used both linear and non-linear features for performing classification and regression analysis. It built simple decision-based rules providing inference from the data which helped in reviewing top features that should be required to make a predictive model for this use case.

   Compared to linear models, decision tree significantly reduced log root mean squared error (RMSE) from 0.74 down to 0.19 with almost 100% accuracy and 5.3% lag between training and test evaluation sets.

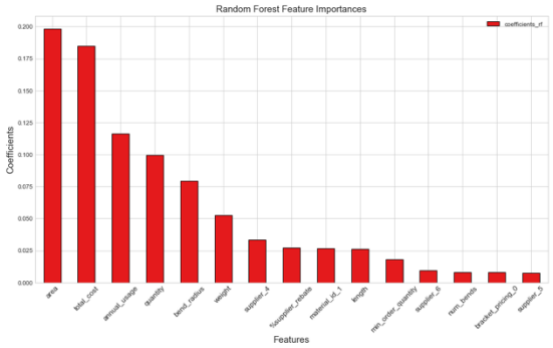
   Such high accuracy without any hyperparameter tuning could be causing decision tree to overfit the model, which is one of the problems in this model, as it continues to build tree-like structure until the last leaf is remaining (i.e. there are no decision nodes left)

7. **Random Forest Regressor**: Unlike decision trees, Random forest consists of a large number of individual decision trees which are majority voted or aggregated to build classification or regression models. The randomness in the model is due to random selection of subset features a.k.a bootstrapping with replacement, and random selection of decision trees of smaller sizes.

   To optimize this model, I used randomized search cross-validation and grid search cross-validation to approximate and finalize hyperparameters. My goal was to identify ideal number of decision trees with maximum depth and minimum node-leaf structure, so that model does not overfit or underfit and provides best mod el performance.

   During hyperparameter tuning, I observed that by not bootstrapping samples, the model used full dataset to build multiple decision trees, which took longer processing time (>30 minutes) and gave 100% accuracy, similar to decision tree model earlier.

   Hence, in trial 2 random and grid search CV, with bootstrapping sampling parameter True, Random Forest predicted top 15 features that made up 89% of supplier pricing prediction with 98% accuracy. The log RMSE was slightly higher from 0.19 to 0.21 and lag between train-test evaluation score lowered from 5.3% to 5.1%.



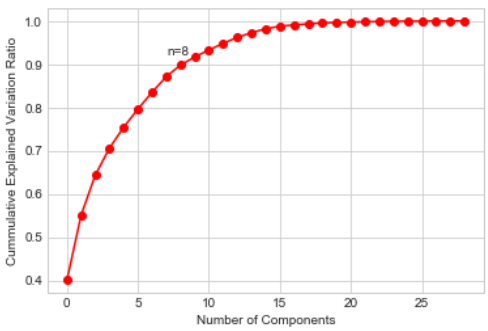
   These top features gained much higher preference than what we obtained from linear regression models earlier, further indicating that linear model coefficients were indeed impacted by residual outliers and influence points.
   Top 15 features:

features
areatotal_costannual_usage
quantitybend_radiusw...coef
ficients_rf
0.894642

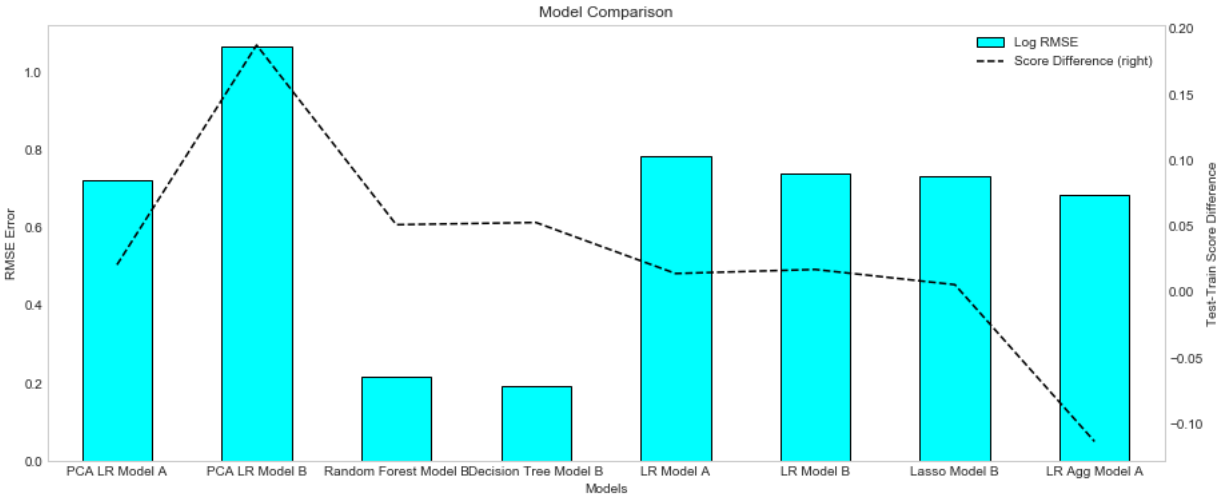
   dtype: object

index	features	coefficients_rf
4	area	0.198479
13	total_cost	0.184663
9	annual_usage	0.116431
11	quantity	0.099507
3	bend_radius	0.079193
8	weight	0.052646
18	supplier_4	0.033473
12	%supplier_rebate	0.027044
66	material_id_1	0.026562
1	length	0.026038
10	min_order_quantity	0.017928
20	supplier_6	0.009315
2	num_bends	0.008023
21	bracket_pricing_0	0.007958
19	supplier_5	0.007382

8. **PCA with Linear Regression using Model A & Model B:** Finally, I tried dimension reduction technique (PCA) by identifying optimal number of principal components that explained maximum variance in the supplier price and passing it through linear regression model, but it resulted in lower training and test evaluation score and high Log RMSE values, similar to aggregation model experimented earlier.



Predictive Model Summary and observations:

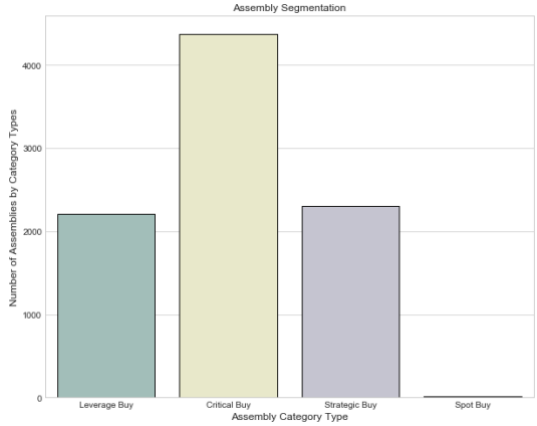
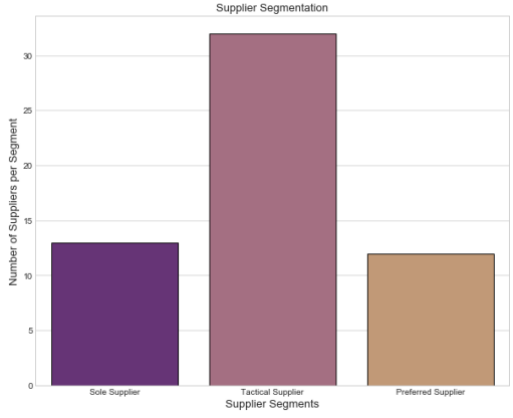


1. **Linear models:** Log RMSE values, as shown by bar, were found comparable with the exception of PCA LR on model B. This is probably because PCA prefers numerical values to find principal components and does not work well with binary encoded data, contained in Model B.

2. **Random Forest:** Overall resulted in good model accuracy 98% after hyperparameter tuning and gave top 15 features which explained 89% variance in predicting supplier pricing. We can use these features to build linear model again using lasso or simple regression model and compare results.

Part 2- Categorizing Tube Assemblies and Suppliers using Clustering:

1. **Pre-Classification**: My goal here was to establish a criteria for segmenting tube assemblies and suppliers. I, anticipated business such as finding cost efficiencies in the form of supplier rebates, building long term contractual relationships and reducing market difficulty so that assemblies can be procured from a low cost technically compliant supplier.
   Below is the pre-segmented distribution of assemblies and suppliers. This will be useful in describing portfolios by business needs and justifying various clusters when we apply unsupervised learning i.e.without target labels, to our model.

Pre-Assembly Segmentation: 4 Categories	Pre-Supplier Segmentation: 3 Categories
	

• Assembly Portfolio Breakdown: ○ Critical Buy 49.1% ○ Strategic Buy 25.9% ○ Leverage Buy 24.8% ○ Spot Buy Negligible	• Supplier Portfolio Breakdown: ○ Tacticle Suppliers 56.1% ○ Sole Suppliers 22.8% ○ Preferred Suppliers 21.1%

2. **Clustering**: After pre-classification was established, I applied log transformation to remove skewness in the data as well as applied standard scaler to ensure model data is on the same unit of scale. This step was crucial in ensuring that meaningful clusters are formed when model is built. After normalizing the data, I built various clustering algorithms such as K-means, Affinity Propagation and few more to group assemblies and suppliers by the business needs.

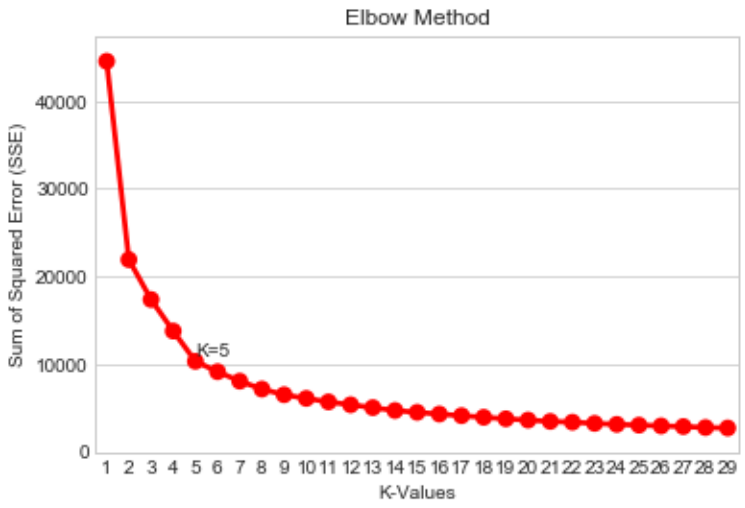
   The objective here was to benchmark these clusters and identify assemblies and suppliers who fully meet pre-set criteria. Another important aspect in analyzing these clusters was evaluating cluster quality. I, used Silhouette score as a measure, which can range between +1 (good cluster) to -1 (poor cluster), as shown in the summary table below.

	Assembly Cluster Quality	Supplier Cluster Quality
Model	Silhouette Score	Silhouette Score
• K-Means	0.46	0.35
• Affinity Propagation	0.41	0.35
• AgglomerativeClustering	0.44	0.34
• SpectralClustering	0.07	0.36
• DBSCAN	0.33	0.14

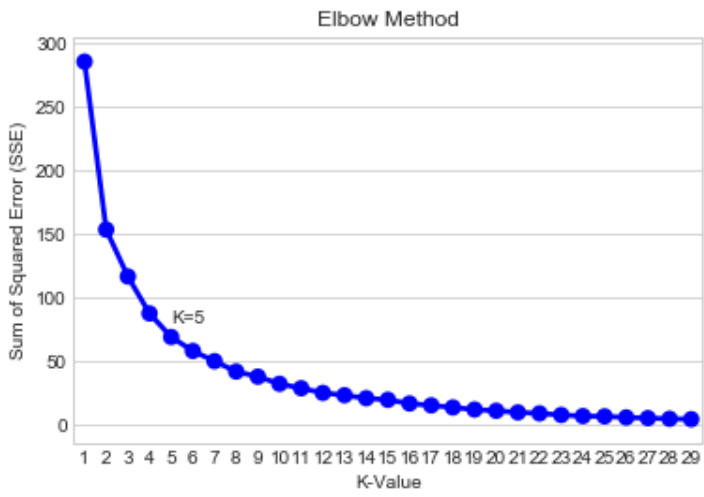
K-means was selected and optimal value for K was selected using elbow method, which is a point where sum of squared error values show marginal decrease with the increase in number of clusters.

In case of supplier clusters, Spectralbiclustering and K-means had comparable silhouette score and less dense cluster quality. Hence, I chose K-means as a common algorithm to compare cluster formation.

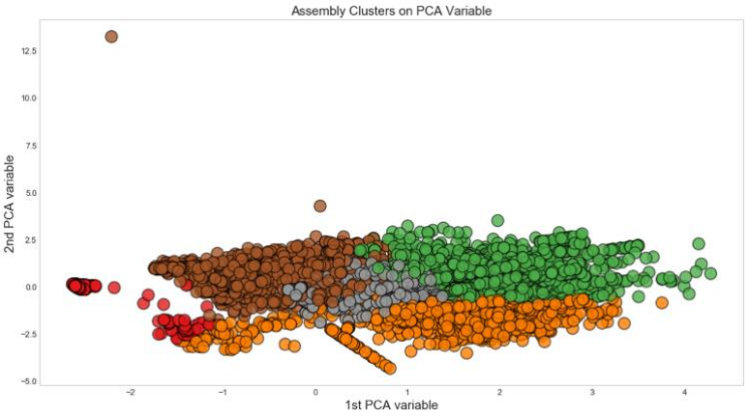
Assembly Clusters	Supplier Clusters
• k-value: 5 clusters	• k-value: 5 clusters



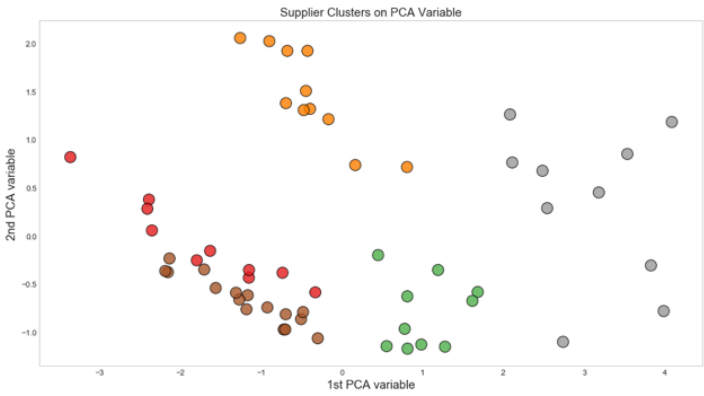
- Quality: Silhouette Score 0.46
  Some overlap between red and brown clusters
  Dense clusters on PCA variables



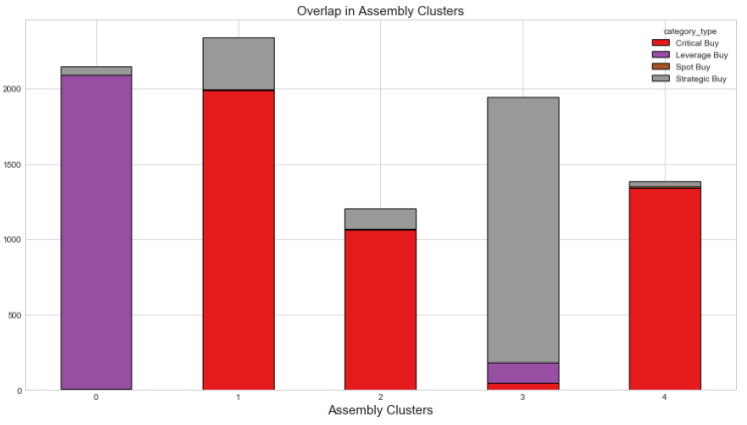
- Quality: Silhouette Score 0.35
  Some overlap between red and brown clusters
  Less Dense Clusters on PCA variables



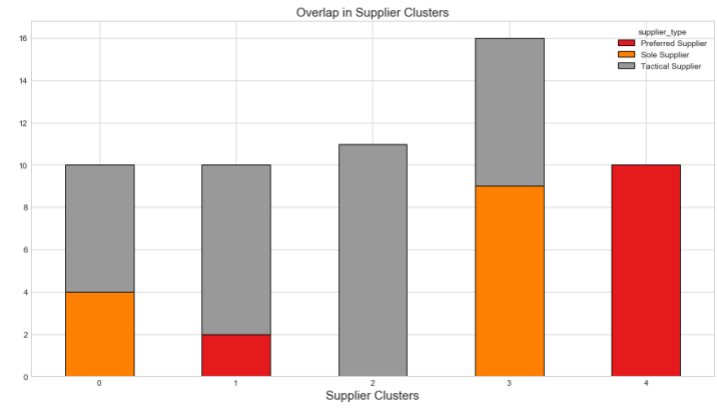
- Overlap: All assembly categories have some Strategic Buy



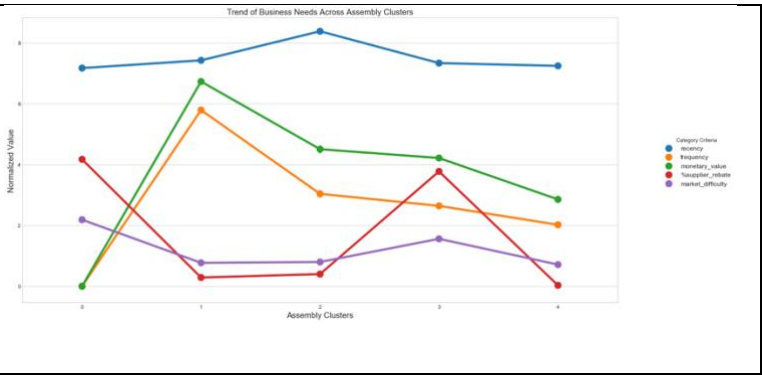
- Overlap: Some Tactical suppliers are Sole as well as Preferred.

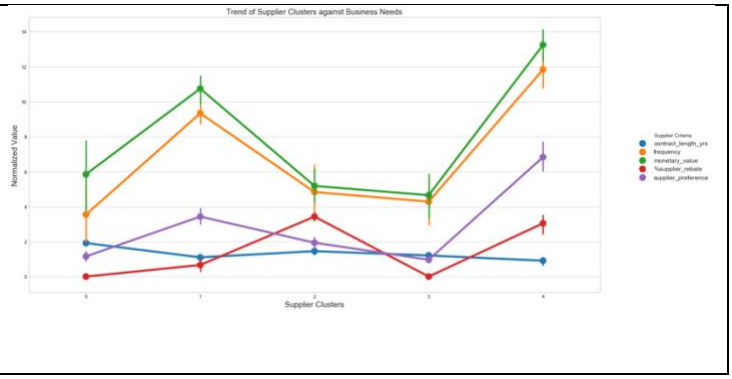


- Benchmark: Cluster# 3
  High rebates fewer suppliers
  Few cluster overlaps



- Benchmark: Cluster# 4
  High rebates high supplier preference
  No overlap





Conclusion:

1. By using Random Forest and K-Means, we were able to establish pricing prediction and benchmark assemblies and supplier clusters from 8,855 unique assemblies and 57 suppliers.
2. There were top 15 features such as tube assembly area, total cost, annual usage, order quantity and more, which explained 89% dependency in predicting supplier pricing.
3. By combining prediction and clustering analysis, business can benefit from estimate supplier pricing based on assembly specifications, demand and order history; identify assembly and supplier portfolios by business needs; and choose group of suppliers that meet business requirements.

Future Possibilities:

1. Monitor Supplier Performance: We can add other features such as on-time delivery, safety statistics, contract compliance and inventory levels to gain further supplier insights.
2. Build Real Time Dashboards: We can connect our model to real time data feed to collect actionable insights on the fly.
3. Further Modeling: We can try other algorithms such as time-series to predict annual supplier pricing.

-----End of report---